UNIVERSIDAD DE ORIENTE

NUCLEO DE ANZOATEGUI

ESCUELA DE INGENIERIA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACION Y SISTEMAS

MODELOS DE OPERACIONES I

****

**PROGRAMACION DINAMICA BASICA**

PROFESORA: INTEGRANTES:

AURELIA TORCASIO. DAVID MENDOZA C.I:18.206.562

MAIRYM NEGRON C.I:18.128.491

EDWARD ROJAS C.I:22.574.155

**INTRODUCCION**

La Programación Dinámica fue desarrollada por Richard Bellman y G B Dantzing. Sus importantes contribuciones sobre esta técnica cuantitativa de toma de decisiones se publicaron en 1957 en un libro del primer autor denominado “Dynamic Programming” (Princeton UniversityPress. Princeton, New Jersey).

Inicialmente a la PD se le denominó programación lineal estocástica ó problemas de programación lineal con incertidumbre. La programación dinámica (PD) determina la solución óptima de un problema de n variables descomponiéndola en n etapas, con cada etapa incluyendo un subproblema de una sola variable. La principal contribución de la PD es el principio de optimalidad, el cual establece que una política óptima consiste de subpolíticas óptimas, un marco de referencia para descomponer el problema en etapas.

La programación dinámica es una técnica que se puede aplicar para resolver muchos problemas de optimización. La mayor parte de las veces, la programación dinámica obtiene soluciones con un avance en reversa, desde el final de un problema hacia el principio con lo que un problema grande y engorroso se convierte en una serie de problemas más pequeños y más tratables.

Así, la programación dinámica se puede definir como una técnica matemática útil que resuelve una serie de decisiones secuenciales, cada una de las cuales afecta las decisiones futuras. Proporciona un procedimiento sistemático para determinar la combinación de decisiones que maximiza la efectividad total. En contraste para el problema de programación dinámica, trata de un enfoque de tipo parcial para la solución de problemas y las ecuaciones específicas que se usan se deben desarrollar para que represente cada situación individual.

**PROGRAMACION DINAMICA**

La programación dinámica es una técnica matemática útil para la toma de una serie de decisiones interrelacionadas. Proporciona un procedimiento sistemático para determinar la combinación óptima de decisiones; comúnmente resuelve el problema en etapas, en donde cada etapa interviene exactamente una variable de decisión. Los cálculos en las diferentes etapas se enlazan a través de cálculos recursivos de manera que se genere una solución óptima factible a todo el problema. Este proceso fue desarrollado por Richard Bellman en 1957.

Sera necesario especificar cada uno de los componentes que lo caracterizan; ya que el procedimiento general de resolución se divide en orden inverso, es decir, comenzando por la última etapa y pasando en cada iteración a la etapa antecesora, el análisis de la primera etapa finaliza con la obtención del último problema.

**PRINCIPIO DE OPTIMALIDAD DE BELLMAN**

La resolución del modelo en forma óptima mediante programación dinámica está garantizada siempre y cuando las soluciones del problema verifiquen el principio de optimalizad de bellman que dice:

“Una solución optima tiene la propiedad de que cualquiera que sea el estado inicial y las decisiones iníciales, las decisiones para las etapas posteriores deben construir una política optima respecto al estado resultante de la primera decisión.”

Otras palabras, las decisiones involucradas desde una etapa en adelante solo dependen del estado inicial de la etapa y no de las decisiones previas.

**ELEMENTOS BASICOS DE UN MODELO DE PROGRAMACION DINAMICA**

Los elementos básicos son:

**Etapas:** la parte del problema que posee un conjunto de alternativas mutuamente excluyentes de las cuales se seleccionara la mejor alternativa. El problema se puede dividir en etapas, en cada una de ellas debe tomarse una decisión que no depende de la etapa anterior si no de la etapa actual. Estas etapas se definirán como: (n=1,2,3,…,i)

**Estado:** es aquel que se refleja la condición de las restricciones que enlazan las etapas. Un estado es el que se refleja la condición de enlace entre las etapas, de tal manera que cuando cada etapa se optimiza por separado, la decisión resultante es factible para el problema completo. Cada estado tiene un número de estados asociados. El conocer este estado es necesario para tomar una decisión óptima. La decisión tomada en esta etapa transforma el estado actual en el estado de la etapa siguiente afectando así las decisiones a tomar. Los estados se denotaran con (Sn).

**RECURSION EN REVERSA Y AVANCE**

**Función Recursiva.**

Dados unos nodos y unos arcos que conectan a estos nodos, el problema de la diligencia intenta encontrar la ruta más corta que conecta un arranque con el nodo final, es decir el destino.

Fi (Xi) representa la distancia o tiempo más corto al nodo (Xi).

Minimizar es el valor mínimo de todas las posibilidades en la primera iteración el valor:

Fi – 1 (Xi - 1)=0

**MODELO DE LA RUTA MÁS CORTA**

Es el camino más corto entre dos nodos de una red, se puede encontrar calculando primero el camino más corto al objetivo desde todos los nodos adyacentes al de partida y después usando estas soluciones para elegir al mejor camino de todos ellos.

Existen dos formas de plantear la formula de recursividad en los problemas de programación dinámica:

1. Recursividad en retroceso: el problema se resuelve partiendo de la última etapa hacia la última.
2. Recursividad en avance: el problema se resuelve partiendo de la primera etapa hacia la última.

La relación recursiva siempre tendrá una de las dos formas siguientes:

**Fi (Xi)= min {d (Xi-1, Xi) + Fi-1 (Xi-1)}**

**Fi (Xi)= max {d (Xi-1, Xi) + Fi-1 (Xi-1)}**

**RESOLUCION DE UN PROBLEMA DE PROGRAMACION DINAMICA**

Para resolver un problema de programación dinámica debemos:

1. **IDENTIFICAR ETAPAS, ESTADOS Y VARIABLES DE DECISION:**

* Cada etapa debe tener asociado una o más decisiones (problema de optimización), cuya dependencia de las decisiones anteriores está dada exclusivamente por las variables de estado.
* Cada estado debe contener toda la información relevante para la toma de decisión asociada al período.
* Las variables de decisión son aquellas sobre las cuales debemos definir su valor de modo de optimizar el beneficio acumulado y modificar el estado de la próxima etapa.

1. **DESCRIPCION DE ECUACION DE RECURRENCIA:**

Deben indicar como se acumula la función de beneficios a optimizar (función objetivo) y como varía las funciones de estado de una etapa a otra.

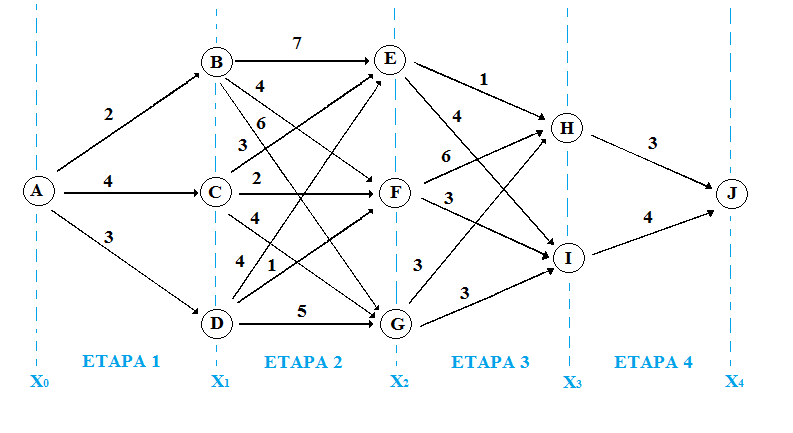
1. **RESOLUCION:**

Se debe optimizar cada subproblema por etapas en función de los resultados de la resolución del subproblema siguiente.

**EJEMPLOS:**

1. **RESUELVA CON PROGRAMACION DINAMICA**

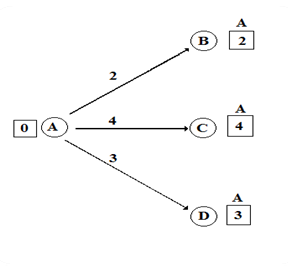
**RECURSIVIDAD EN AVANCE “METODO GRAFICO":**

****

Un viajero desea trasladarse desde la ciudad “A” hasta la ciudad “J” empleando la menor cantidad de tiempo posible. En el siguiente grafico se expresan los posibles caminos y sus tiempos asociados en horas.

Especificando las etapas y sus variables de estado con sus resultados en cada una.

**ETAPA 1:**



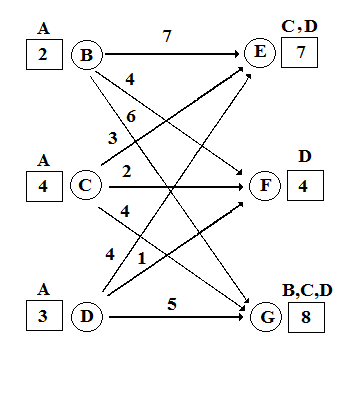
**Distancia más Corta:**

**Nodo B: 2 ( Nodo A )**

**Nodo C: 4 ( Nodo A )**

**Nodo D: 3 ( Nodo A )**

**ETAPA 2:**

****

**Distancia más Corta:**

**Nodo E:**

**B – E → (7 + 2) = 9**

**C – E → (3 + 4) = 7**

**D – E → (4 + 3) = 7**

**Nodo F:**

**B – F → (4 + 2) = 6**

**C – F → (2 + 4) = 6**

**D – F → (1 + 3) = 4**

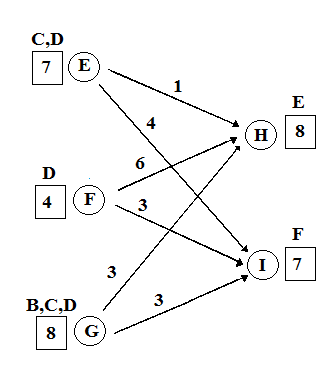
**Nodo G:**

**B – G → (6 + 2) = 8**

**C – G → (4 + 4) = 8**

**D – G → (5 + 3) = 8**

**ETAPA 3:**

****

**Distancia más Corta:**

**Nodo H:**

**E – H → (1 + 7) = 8**

**F – H → (6 + 4) = 10**

**G – H → (3 + 8) = 11**

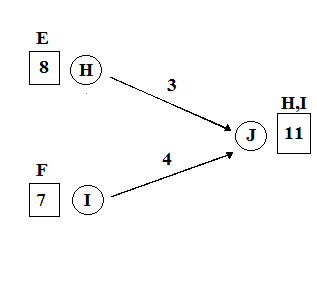
**Nodo I:**

**E – I → (4 + 7) = 11**

**F – I → (3 + 4) = 7**

**G – I → (3 + 8) = 11**

**ETAPA 4:**

****

**Distancia más Corta:**

**Nodo J:**

**H – J → (3 + 8) = 11**

**I – J → (4 + 7) = 11**

**Rutas:**

**{A – C – E – H – J }→ {11 Horas}**

**{A – D – E – H – J}→ {11 Horas}**

**{A – D – F – I – J}→ {11 Horas}**

1. **RECURSIVIDAD EN AVANCE DEL “METODO TABULAR”:**

**Método Tabular.**

**Recursividad en Reversa:**

**Fi (Xi)=máx. {d (Xi, Xi+1) + F (i+1) X (i+1) }**

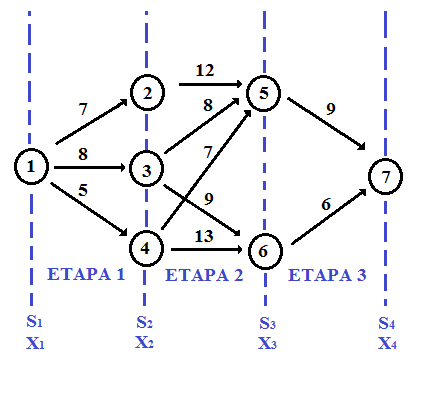
**min**

**Recursividad en avance:**

**Fi (Xi)= máx. {d ( Xi-1, Xi) + Fi-1( Xi-1) }**

**min**

Supongamos que deseamos seleccionar la ruta por carretera mas corta entre dos ciudades. La red proporciona las posibles rutas entre la ciudad de inicio en el nodo 1 y la ciudad destino en el nodo 7. Las rutas pasan por ciudades intermedias designadas por los nodos 2 a 6.

****

**Etapa 1: (i=1); Xo=1**

**F1(X1)=min {a (X0, X1) + F0 (X0) = 0}**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **d(X0,X1) + F0(X0)** | **Solución** | **Optima** |
| **X1** | **X0= 1** | **F1(X1)** | **X0** |
| **2** | **7+0=7** | **7** | **1** |
| **3** | **8+0=8** | **8** | **1** |
| **4** | **5+0=5** | **5** | **1** |

**Etapa 2: (i=2); X1=2, 3,4**

**F2(X2)=min {a (X1, X2) + F1 (X1)}**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **d(X1,X2) + F1(X1)** | | | **Solución** | **Optima** |
| **X2** | **X1= 2** | **X1=3** | **X1=4** | **F2(X2)** | **X1** |
| **5** | **12+7=17** | **8+8=16** | **7+5=12** | **12** | **4** |
| **6** | **−** | **9+8=17** | **13+5=18** | **17** | **3** |
|  | | | | | |

**Etapa 3: (i=3); X2=5,6**

**F3(X3)=min {a (X2, X3) + F2 (X2)}**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **d(X2,X3) + F2(X2)** | | **Solución** | **Optima** |
| **X3** | **X2= 5** | **X2=6** | **F3(X3)** | **X2** |
| **7** | **9+12=21** | **6+17=23** | **21** | **5** |
|  | | | |  |

**Rutas:**

**X3=7→ X2=5 →X1=4 → X0=1**

**RECURSIVIDAD EN REVERSA DEL “METODO TABULAR”:**

**Etapa 3:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **d(X3,X4) + F4(X4)** | **Solución** | **Optima** |
| **X3** | **X4=7** | **F3(X3)** | **X4** |
| **5** | **9+0=9** | **9** | **7** |
| **6** | **6+0=6** | **6** | **7** |

**F3(X3)=min {a (X3, X4) + F4 (X4) = 0}**

**Etapa 2:**

**F2(X2)=min {a (X2, X3) + F3 (X3)}**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **d(X2,X3) + F3(X3)** | | **Solución** | **Optima** |
| **X2** | **X3= 5** | **X3=6** | **F2(X2)** | **X3** |
| **2** | **12+9=21** | **−** | **21** | **5** |
| **3** | **8+9=17** | **9+6=15** | **15** | **6** |
| **4** | **7+9=16** | **13+6=19** | **16** | **5** |

**Etapa 1:**

**F1(X1)=min {a (X1, X2) + F2 (X2)}**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **d(X1,X2) + F2(X2)** | | | **Solución** | **Optima** |
| **X1** | **X2= 2** | **X2=3** | **X2=4** | **F1(X1)** | **X2** |
| **1** | **7+21=28** | **8+15=23** | **5+16=21** | **21** | **4** |
|  | | | | | |

**Rutas:**

**X1=1 → X2=4 → X3=5 → X4=7**

**BIBLIOGRAFIA**

* **INVESTIGACION DE OPERACIONES, AUTOR TAHA. https://jrvargas.files.wordpress.com/2009/01/investigacic3b3n-de-operaciones-9na-edicic3b3n-hamdy-a-taha-fl.pdf**